Stochastic Online Scheduling on Parallel Machines

Stochastic Online Scheduling on Parallel Machines*

Nicole Megow[1], Marc Uetz[2], and Tjark Vredeveld[3]

[1] Technische Universität Berlin, Institut für Mathematik, Strasse des 17. Juni 136, 10623 Berlin, Germany. E-mail: nmegow@math.tu-berlin.de
[2] Maastricht University, Department of Quantitative Economics, P.O.Box 616, 6200 MD Maastricht, The Netherlands. E-mail: m.uetz@ke.unimaas.nl
[3] Konrad-Zuse-Zentrum für Informationstechnik Berlin, Department Optimization, Takustr. 7, 14195 Berlin, Germany. E-mail: vredeveld@zib.de

Abstract. We consider a non-preemptive, stochastic parallel machine scheduling model with the goal to minimize the weighted completion times of jobs. In contrast to the classical stochastic model where jobs with their processing time distributions are known beforehand, we assume that jobs appear one by one, and every job must be assigned to a machine online. We propose a simple online scheduling policy for that model, and prove a performance guarantee that matches the currently best known performance guarantee for stochastic parallel machine scheduling. For the more general model with job release dates we derive an analogous result, and for NBUE distributed processing times we even improve upon the previously best known performance guarantee for stochastic parallel machine scheduling. Moreover, we derive some lower bounds on approximation.

1 Introduction

Non-preemptive parallel machine scheduling to minimize the weighted completion times of jobs, $P||\sum w_jC_j$ in the three-field notation of Graham et al. [6], is one of the classical problems in scheduling theory. This problem plays a role whenever many jobs must be processed on a limited number of machines, with typical applications, e.g., in parallel computing [2] or compiler optimization [3]. The main characteristic of the model of *stochastic scheduling* is the fact that the processing times of jobs are subject to fluctuations, and become known only upon completion of the jobs. Their respective distributions are assumed to be given beforehand. This usually requires the notion of *scheduling policies* instead of simple schedules.

Stochastic scheduling. Stochastic machine scheduling models have been addressed mainly since the 1980s [4]. Let us briefly recall the concept of a scheduling policy as introduced by Möhring et al. [10]. Roughly spoken, at any time t, such a

* Research partially supported by the DFG Research Center "Mathematics for key technologies" (FZT 86) in Berlin.

policy specifies which action to perform, in particular which jobs to start at t. In order to decide, it may utilize the complete information contained in the partial schedule up to time t. However, it must not utilize any information about the future. An *optimal* scheduling policy is one that minimizes the objective function value in expectation. Notice that, in general, a scheduling policy need not yield a fixed assignment of jobs to machines.

With the exception of the papers by Weiss [18,19], the first approximation algorithms for stochastic machine scheduling have been derived only recently [11,13,14,16]. In the papers [11,14], the expected performance of a linear programming based list scheduling policy is compared against the expected performance of an optimal scheduling policy. The results are constant-factor approximations for problems with or without release dates [11], and also with precedence constraints [14]. The approach is based upon the solution of linear programming relaxations, and for the models with release dates or precedence constraints, their solutions are used in order to define corresponding list scheduling policies. Recently, another type of analysis has been pursued by Steger et al. in the papers [13,16], where the expected *ratio* of the performance of the (W)SEPT rule[4] over the optimum solution is analyzed. This approach may indeed have advantages over the previous approach, namely in terms of averaging over different realizations of processing times, and we refer to [13,16] for a discussion. One of the main differences, however, is the fact that it uses a comparison against the off-line optimum, whereas the approach in [11,14] compares against the on-line optimum. Nevertheless, restricted to models without release dates or precedence constraints, constant-factor approximation results for the expected ratio have been obtained for the (W)SEPT rule on parallel machines [13,16].

Stochastic online scheduling. In this paper, we follow the approach taken by Möhring et al. [11]. In other words, we compare the expected outcome of a certain scheduling policy against the expected outcome of an optimal scheduling policy. In contrast to the previously mentioned work on stochastic scheduling, however, we consider a model where jobs have to be assigned to machines online. More precisely, jobs j are presented to the scheduler one by one, with their weights w_j and expected processing times $\mathbb{E}[P_j]$, and without knowledge about jobs that might appear in the future, or their number, they must be assigned to a machine. This assignment cannot be revised later. Once all jobs have been assigned this way, there is freedom concerning the scheduling of jobs on every single machine; of course, still within the restrictions that jobs must not be preempted, and that their actual processing times become known only upon completion. For convenience, let us denote this model Sos, for stochastic online scheduling.

Discussion of the model. As a matter of fact, the solution of LP relaxations is crucial for the work of Möhring et al. [11] or Skutella and Uetz [14]. For models

[4] In the WSEPT rule, jobs are scheduled greedily in the order of non-increasing ratios $w_j/\mathbb{E}[P_j]$, where w_j is the weight of job j, P_j its processing time distribution, and $\mathbb{E}[P_j]$ is its expectation. For unit weights, this equals SEPT; shortest expected processing time first.

with release dates or precedence constraints, optimal LP solutions are not only required for the purpose of analysis, but also to define the corresponding list scheduling policies. In order to set up these LP relaxations, it is required to know beforehand the set of jobs, their expected processing times $\mathbb{E}[P_j]$, as well as a uniform upper bound Δ on the squared coefficient of variation of all processing times distributions

$$\mathrm{CV}[P_j]^2 = \mathrm{Var}[P_j]/\mathbb{E}[P_j]^2 \leq \Delta \quad \text{for all jobs } j\,.$$

One critique of this approach is the fact that in practical applications, parts of this data might not be available. Even worse, in an online setting there is no knowledge about jobs that might appear in the future. In that case, algorithms that first require the solution of sophisticated LP relaxations might be useless.

The Sos model as proposed in this paper can be seen as a first step in the direction of simpler, combinatorial algorithms for models with stochastic processing times. It is a two-phase model, where the first phase consists of an online assignment of jobs to machines. In this phase, whenever a job j is presented to the scheduler, the only information that is disclosed is its weight w_j and its expected processing time $\mathbb{E}[P_j]$. In the model with release dates, it is also the release time r_j. The second phase consists of the actual process of scheduling the jobs over time, processing times being realized according to the respective distributions. Yet, we compare the expected outcome of the online stochastic scheduling policy to the expected outcome of an optimal scheduling policy, according to the definition of general scheduling policies by Möhring et al. [10].

In comparison to classical online models, we make two remarks. First, like in classical online optimization, the adversary in the Sos model may choose an arbitrary sequence of jobs in the first phase. These jobs are assumed to be stochastic, with corresponding processing time distributions (deterministic jobs being a special case). However, in the second phase, the actual processing times are realized according to the exogenous probability distributions, thus they are not under control of the adversary. Moreover, given the exogenously controlled processing times, the best the adversary can do is in fact to use an optimal stochastic scheduling policy. In this view, our model indeed incorporates some of the ideas by Koutsoupias and Papadimitriou in [8].

Results and methodology. We derive worst case performance guarantees for the expected performance of very simple, combinatorial online scheduling policies for models with and without release dates. For the model without release dates, $\mathrm{P}|\,|\mathbb{E}[\sum w_j\,C_j]$, this is a performance guarantee of

$$1 + \frac{(\Delta+1)(m-1)}{2m}\,,$$

matching the previously best known performance guarantee of [11] for the performance of the WSEPT rule. Note, however, that this bound holds even though we use a restricted scheduling policy that first has to assign jobs to machines online, without knowledge of the jobs to come. For the model with release dates,

$P|r_j|\mathbb{E}\left[\sum w_j C_j\right]$ we prove a more complicated performance guarantee for a class of processing time distributions that we call δ-NBUE. They generalize NBUE distributions[5], which are contained as a special case. For NBUE distributions, we obtain a performance bound strictly less than

$$\frac{5+\sqrt{5}}{2} - \frac{1}{2m},$$

where $(5+\sqrt{5})/2 \approx 3.62$. Thereby, we improve upon the previously best known performance guarantee of $4 - 1/m$ for NBUE distributions, which was derived for an LP based list scheduling policy [11]. Again, notice that this improved bound holds even though we use a restricted policy that first has to assign jobs to machines online, without knowledge of the jobs to come.

Our results are achieved by the following, quite simple SOS policy. Once the jobs have been assigned to the machines, we assume that on every machine the jobs are processed in the WSEPT order[6]. To make the online decisions on machine assignments in the first phase, at any time when a job is presented, we assign it to that machine where it causes the minimal increase in total expected objective value; given the jobs that have been assigned so far, and given that the jobs on each machine will be scheduled in WSEPT order. Intuitively, the reason why we can recover (or improve, respectively) the previous best known results in stochastic machine scheduling is the following: On the one hand, we restrict the full power of scheduling policies by fixing machine assignments beforehand. On the other hand, it is precisely this fixed machine assignment, together with an averaging argument over the number of machines, that allows an improvement in the analysis in comparison to general scheduling policies. We mention that, to obtain our results, we in fact utilize one of the LP based lower bounds of [11].

2 Model definition, notation, and preliminaries

Let n be the number of jobs, index $j \in \{1,\dots,n\}$ denote a job, with associated weight w_j and processing time distribution P_j. By $\mathbb{E}[P_j]$ we denote its expected processing time, and p_j denotes a particular realization of P_j. In the model with release dates, r_j denotes the earliest point in time when job j can be started. Given a schedule of start times $S_1,\dots,S_n$ for a particular realization $p = (p_1,\dots,p_n)$ of processing times, $C_j = S_j + p_j$ is the completion time of job j, $j = 1,\dots,n$. Each job must be processed non-preemptively, on any of the m machines, and each machine can process at most one job at a time. The goal is to find a scheduling policy that minimizes the expected value of the weighted completion times of jobs, $\mathbb{E}\left[\sum w_j C_j\right]$.

A scheduling policy eventually yields a feasible m-machine schedule for each realization p of the processing times. For a given policy, denoted by Π, let $S_j^{\Pi}(p)$

[5] A distribution X is called NBUE, new better than used in expectation, if $\mathbb{E}[X - t \mid X > t] \leq \mathbb{E}[X]$ for all $t > 0$.

[6] In the case with release dates, this is in fact a modified version of the WSEPT order, that will be explained later.

and $C_j^{\Pi}(p)$ denote the start and completion times of job j for a given realization p, and let $S_j^{\Pi}(P)$ and $C_j^{\Pi}(P)$ denote the associated random variables. Unless there is danger of ambiguity, we also write S_j and C_j, for short. We let

$$\mathbb{E}\left[Z^{\Pi}\right] = \mathbb{E}\left[\sum\nolimits_j w_j\, C_j^{\Pi}(P)\right]$$

denote the expected performance of a scheduling policy Π. Then, if OPT is an optimal scheduling policy according to the most general definition of stochastic scheduling policies in [10], we say that a policy Π is a *ρ–approximation* if, for some $\rho \geq 1$,

$$\mathbb{E}\left[Z^{\Pi}\right] \leq \rho\, \mathbb{E}\left[Z^{\mathrm{OPT}}\right].$$

We assume that the jobs are presented to the scheduler one by one, in the order $1, \ldots, n$. However, the number of jobs n is not known before. When a job is presented, the scheduler is informed about its weight w_j and its expected processing time $\mathbb{E}[p_j]$ (in the case with release dates, also its release date r_j). When job j appears, it must be assigned to a machine $i \in \{1, \ldots, m\}$ immediately, and this decision must not be revised later. For a given job $j \in W$, and a given subset of jobs W, let us define by $H(j)$ the jobs in W that have a higher priority in the WSEPT ordering, that is

$$H(j) = \left\{ k \in W \mid \frac{w_k}{\mathbb{E}[P_k]} \geq \frac{w_j}{\mathbb{E}[P_j]} \right\}.$$

Notice that, by convention, $H(j)$ contains job j, too. Accordingly, define

$$L(j) = \left\{ k \in W \mid \frac{w_k}{\mathbb{E}[P_k]} < \frac{w_j}{\mathbb{E}[P_j]} \right\}$$

as those jobs that have lower priority in the WSEPT order.

It is clear that the online scheduling policies for the Sos model can in fact be interpreted as a subclass of stochastic scheduling policies in general. This because, assuming a classical input for a stochastic scheduling problem where all (stochastic) input data is disclosed at the outset, the only thing we require in the Sos model is a fixed assignment of jobs to machines beforehand. Therefore, the expected performance of an optimal Sos policy is no less than the expected performance of an optimal policy for a corresponding classical stochastic problem. (The latter being defined by the input after the online phase.) Hence, lower bounds on the expected value of an optimal policy known from stochastic scheduling carry over to the online setting. We crucially exploit that fact, and will utilize the following lower bound on the expected performance $\mathbb{E}\left[Z^{\mathrm{OPT}}\right]$ of an optimal stochastic scheduling policy. It is a generalization of a lower bound by Eastman et al. [5] for the deterministic setting.

Lemma 1 (Möhring et al. [11]). *For any instance of* $\mathrm{P}|\,|\mathbb{E}[\sum w_j\, C_j]$*, we have that*

$$\mathbb{E}[Z^{\mathrm{OPT}}] \geq \sum_j w_j \sum_{k \in H(j)} \frac{\mathbb{E}[P_k]}{m} \;-\; \frac{(m-1)(\Delta-1)}{2m} \sum_j w_j \mathbb{E}[P_j]\,,$$

where Δ bounds the squared coefficient of variation of the processing times, that is, $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2 \leq \Delta$ *for all jobs* $j = 1, \ldots, n$ *and some* $\Delta \geq 0$.

This lemma indeed plays a crucial role in achieving performance guarantees for the SOS policies presented in the following sections. Clearly, the claim of Lemma 1 also applies to the setting with release dates $\mathrm{P}|r_j|\mathbb{E}[\sum w_j C_j]$.

3 Lower bounds on approximation

The requirement of a fixed assignment of jobs to machines beforehand may be interpreted as ignoring the additional information on the outcome of the stochastic process (that is, the actual realization of processing times), at least with respect to assignments of jobs to machines. In the following, we therefore give a lower bound on the expected performance $\mathbb{E}\left[Z^{\mathrm{FIX}}\right]$ of an optimal stochastic scheduling policy FIX that assigns jobs to machines beforehand. A fortiori, this lower bound holds for the best possible SOS policy, too.

Theorem 1. *For stochastic parallel machine scheduling with unit weights and i.i.d. exponential processing times,* $\mathrm{P}|p_j \sim \exp(1)|\mathbb{E}[\sum C_j]$*, there exist instances such that*

$$\mathbb{E}\left[Z^{\mathrm{FIX}}\right] \geq 3(\sqrt{2}-1) \cdot \mathbb{E}\left[Z^{\mathrm{OPT}}\right] - \varepsilon\,,$$

for any $\varepsilon > 0$*. Here,* $3(\sqrt{2}-1) \approx 1.24$*. Hence, no policy that uses fixed assignments of jobs to machines can perform better in general.*

Notice that the Theorem is formulated for the special case of exponentially distributed processing times. Stronger bounds could probably be obtained for arbitrary distributions. However, since our performance guarantees, as in [11], will depend on the coefficient of variation of the processing times, we are particularly interested in lower bounds for classes of distributions where this coefficient of variation is small. The coefficient of variation of exponentially distributed random variables equals 1. For example, for the case of $m = 2$ machines, we get a lower bound of $8/7 \approx 1.14$, and for that case our performance bound equals $2 - 1/m = 1.5$.

Proof (of Theorem 1). For simplicity, we will prove a slightly worse lower bound. Let us consider an instance with m machines and $n = m + \lceil m/2 \rceil$ exponentially distributed jobs, $p_j \sim \exp(1)$. The optimal stochastic scheduling policy is SEPT, shortest expected processing time first [1,20], and the expected performance is (see, e.g., [17, Cor. 3.5.17])

$$\mathbb{E}\left[Z^{\mathrm{OPT}}\right] = \mathbb{E}\left[Z^{\mathrm{SEPT}}\right] = \sum_j \mathbb{E}\left[C_j^{\mathrm{SEPT}}\right] = m + \sum_{j=m+1}^{n} \frac{j}{m}\,.$$

The best fixed assignment policy assigns 2 jobs each to $\lceil m/2 \rceil$ of the machines, and 1 job each to $\lfloor m/2 \rfloor$ of the machines. Hence, there are m jobs

with $\mathbb{E}[C_j] = 1$, and $\lceil m/2 \rceil$ jobs with $\mathbb{E}[C_j] = 2$. The expected performance for the best fixed assignment policy FIX is

$$\mathbb{E}\left[Z^{\mathrm{FIX}}\right] = \sum_j \mathbb{E}\left[C_j^{\mathrm{FIX}}\right] = m + 2 \cdot \lceil m/2 \rceil .$$

For small values $m = 2, 3, 4 \ldots$, we calculate $\mathbb{E}\left[Z^{\mathrm{FIX}}\right]/\mathbb{E}\left[Z^{\mathrm{OPT}}\right] = 8/7$, $7/6, 32/27, \ldots$. It is easy to see that

$$\frac{\mathbb{E}\left[Z^{\mathrm{FIX}}\right]}{\mathbb{E}\left[Z^{\mathrm{OPT}}\right]} = \frac{16m^2}{13m^2 + \mathrm{o}(m^2)} ,$$

and for $m \to \infty$ we get a lower bound of $16/13 \approx 1.23$. Now the claim of the theorem follows along the same lines if we redefine the number of jobs as $n = m + \lceil \sqrt{2}m \rceil$. □

4 Stochastic online scheduling

We next define a stochastic online scheduling policy for the problem without release dates, $\mathrm{P}|\,|\mathbb{E}\left[\sum w_j C_j\right]$. The basic idea is simple: any job j, once it appears, will be assigned to the machine where it causes the minimal increase in expected objective value (given that jobs $1, \ldots, j-1$ have been assigned already). In order to be able to do that, we first need to specify how the jobs will be scheduled on every single machine. We introduce a final bit of notation, letting M_i denote all jobs that are assigned to machine i, and letting $M_i(j) = \{k < j \mid k \in M_i\}$ denote the subset of jobs assigned to a machine i before some job j appears[7].

WSEPT (weighted shortest expected processing time first)
On each machine i, schedule the jobs M_i in non-decreasing order of their ratios of weight over expected processing time $w_j/\mathbb{E}[P_j]$.

This policy is known to be optimal for (stochastic) machine problems on a single machine, $1|\,|\mathbb{E}\left[\sum w_j C_j\right]$, by results of Smith and Rothkopf, respectively [15,12]. Now we can define the MinIncrease policy as follows.

MinIncrease
When a job j is presented to the scheduler, it is assigned to the machine i that minimizes the expression

$$\mathrm{incr}\,(j,i) = w_j \cdot \sum_{k \in H(j) \cap M_i(j)} \mathbb{E}[P_k] \quad + \quad \mathbb{E}[P_j] \cdot \sum_{k \in L(j) \cap M_i(j)} w_k \quad + w_j \mathbb{E}[P_j] .$$

In fact, given that WSEPT is used on each machine, MinIncrease just chooses the machine where job j causes the least increase in expected performance.

[7] Recall that we assume a numbering of the jobs in the order in which they appear in the online sequence; hence $k < j$ denotes jobs k that appeared earlier than job j.

Theorem 2. *Consider the stochastic online scheduling problem on parallel machines,* $\mathrm{P}|\,|\mathbb{E}\left[\sum w_j C_j\right]$. *Given that* $\mathrm{Var}[P_j]/\mathbb{E}\left[P_j\right]^2 \le \Delta$ *for all jobs* j *and some constant* $\Delta \ge 0$, *the* MinIncrease *policy is a* ρ*–approximation, where*

$$\rho = 1 + \frac{(\Delta+1)(m-1)}{2m}\,.$$

Proof. Denote by $\mathbb{E}\left[\mathrm{incr}\,(j)\,\right]$ the increase in the expected objective value caused by fixing the assignment of a job j using MinIncrease. Since MinIncrease chooses the machine i on which j causes the least expected increase, the expected increase is

$$\begin{aligned}
\mathbb{E}\left[\mathrm{incr}\,(j)\,\right] &= \min_i\{\mathbb{E}\left[\mathrm{incr}\,(j,i)\,\right]\} \\
&= \min_i\left\{ w_j \sum_{k\in H(j)\cap M_i(j)} \mathbb{E}\left[P_k\right] \;+\; \mathbb{E}\left[P_j\right] \sum_{k\in L(j)\cap M_i(j)} w_k \right\} + w_j\mathbb{E}\left[P_j\right] \\
&\le \frac{1}{m}\left(w_j \sum_{k\in H(j),k<j} \mathbb{E}\left[P_k\right] \;+\; \mathbb{E}\left[P_j\right] \sum_{k\in L(j),k<j} w_k \right) + w_j\mathbb{E}\left[P_j\right]\,,
\end{aligned}$$

where the inequality holds because the least expected increase is not more than the average expected increase over all machines. By summing up these quantities over all jobs we obtain the expected performance $\mathbb{E}\left[Z^{\mathrm{MI}}\right]$ of the MinIncrease policy.

$$\begin{aligned}
\mathbb{E}\left[Z^{\mathrm{MI}}\right] &= \sum_j \mathbb{E}\left[\mathrm{incr}\,(j)\,\right] \\
&\le \frac{1}{m}\sum_j\left(w_j \sum_{k\in H(j),k<j} \mathbb{E}\left[P_k\right] \;+\; \mathbb{E}\left[P_j\right] \sum_{k\in L(j),k<j} w_k \right) + \sum_j w_j\mathbb{E}\left[P_j\right] \\
&= \frac{1}{m}\sum_j w_j \sum_{k\in H(j)} \mathbb{E}\left[P_k\right] \;+\; \frac{m-1}{m}\sum_j w_j\mathbb{E}\left[P_j\right]\,,
\end{aligned}$$

where the last equality holds by index rearrangement, since

$$\sum_j \mathbb{E}\left[P_j\right] \sum_{k\in L(j),k<j} w_k \quad = \quad \sum_j w_j \sum_{k\in H(j),k>j} \mathbb{E}\left[P_k\right]\,.$$

Now, we plug in the inequality of Lemma 1, and using the trivial fact that $\sum_j w_j\mathbb{E}\left[P_j\right]$ is a lower bound for the expected performance $\mathbb{E}\left[Z^{\mathrm{OPT}}\right]$ of an optimal policy, we obtain

$$\begin{aligned}
\mathbb{E}\left[Z^{\mathrm{MI}}\right] &\le \mathbb{E}\left[Z^{\mathrm{OPT}}\right] + \frac{(\Delta-1)(m-1)}{2m}\sum_j w_j\mathbb{E}\left[P_j\right] + \frac{m-1}{m}\sum_j w_j\mathbb{E}\left[P_j\right] \\
&\le \left(1 + \frac{(\Delta+1)(m-1)}{2m}\right)\cdot \mathbb{E}\left[Z^{\mathrm{OPT}}\right]\,. \qquad \square
\end{aligned}$$

This performance guarantee matches the currently best known performance guarantee for the classical stochastic setting, which was derived for the performance of the WSEPT rule in [11]. The WSEPT rule, however, requires the knowledge of all jobs with their weights w_j and expected processing times $\mathbb{E}[P_j]$ at the outset. In contrast, the MinIncrease policy decides on machine assignments online, without any knowledge of the jobs to come. Finally, it is worthy to note that simple instances show that these two policies are indeed different.

Lower bounds for MinIncrease. The lower bound on the performance ratio for *any* fixed assignment policy given in Theorem 1 holds for the MinIncrease policy, too. Hence, in general, MinIncrease cannot be better than 1.24-approximative. We can strengthen the lower bound via more sophisticated instances, but the computations of the optimal values become unpleasant. We next give an instance for $m = 2$ machines.

Example 1. We are given 6 jobs with exponentially distributed processing times such that $\mathbb{E}[P_1] = \mathbb{E}[P_4] = 1, \mathbb{E}[P_2] = \mathbb{E}[P_5] = k$ and $\mathbb{E}[P_3] = \mathbb{E}[P_6] = 2k$, for some fixed k. The jobs appear in order of their indices in the online sequence.

Without going into further details, it turns out that the expected performance of the MinIncrease policy is $6 + 9k$, and the expected performance of an optimal scheduling policy is $5 + k(167/24 + 7/(1 + k) + 1/(2 + 4k))$. For $k \to \infty$, this yields a lower bound of $216/167 \approx 1.29$, whereas Theorem 2 yields a performance guarantee of 1.5.

For less restricted probability distributions, i.e., non-exponential and with larger coefficients of variation, we obtain a lower bound of $3/2$ on the expected performance of MinIncrease relative to an optimal scheduling policy. However, this is less meaningful compared to the performance bound of Theorem 2, which depends on an upper bound Δ on the squared coefficient of variation. We skip the details.

5 Stochastic online scheduling with release dates

In this section we consider the problem of stochastic online scheduling on parallel machines where jobs have release dates. As the optimal single machine scheduling policy is unknown to date for this problem, we analyze the expected performance of the MinIncrease policy which runs the following single machine scheduling policy.

> **α-Shift-WSEPT**
> Modify the release date r_j of each job j such that $r'_j = \max\{r_j, \alpha\,\mathbb{E}[P_j]\}$, for some fixed $0 < \alpha \leq 1$. At any time t, when the machine is idle, start the job with the highest priority in the WSEPT order among all available jobs (respecting the modified release dates).

In the deterministic (online) setting, this policy was proposed for parallel machines in [9]. For the analysis of this policy, we restrict ourselves to random

variables that we call δ-NBUE. This is a generalization of NBUE random variables.

Definition 1 (δ-NBUE). *A non-negative random variable X is δ-NBUE if, for $\delta \geq 1$,*

$$\mathbb{E}\,[X - t \,|\, X > t\,] \;\leq\; \delta\,\mathbb{E}\,[X] \qquad \text{for all } t \geq 0.$$

Ordinary NBUE distributions are by definition 1-NBUE. For a NBUE random variable X, Hall and Wellner [7] showed that the (squared) coefficient of variation is bounded by 1, that is, $\mathrm{Var}[X]/\mathbb{E}\,[X]^2 \leq 1$. From their work, it also follows that, if X is δ-NBUE, then $\mathrm{Var}[X]/\mathbb{E}\,[X]^2 \leq 2\delta - 1$. Examples of ordinary NBUE (or 1-NBUE) distributions are exponential, Erlang, uniform, or Weibull distributions (with shape parameter at least 1). We next derive an upper bound on the expected completion time of a job, $\mathbb{E}\,[C_j]$, when scheduling jobs on a single machine according to the α-Shift-WSEPT policy. This bound is used later to analyze the expected performance of MinIncrease.

Lemma 2. *Let all processing times be δ-NBUE. Then the expected completion time of job j for α-Shift-WSEPT on a single machine can be bounded by*

$$\mathbb{E}\,[C_j] \leq (1 + \delta/\alpha)\, r'_j + \sum_{k \in H(j)} \mathbb{E}\,[P_k]\ .$$

Proof. We consider some job j. Let us denote by $\mathcal{B}$ the event that the machine is busy processing some job at time r'_j, and let us denote by $\mathcal{I}$ the complement of $\mathcal{B}$, namely that the machine is idle (or just finished processing some job) at time r'_j. Under the condition $\mathcal{I}$ it could still be that there are higher priority jobs $k \in H(j) \setminus \{j\}$ available at time r'_j, but in any case the expected start time of job j can be postponed by at most

$$\sum_{k \in H(j) \setminus \{j\}} \mathbb{E}\,[P_k \mid \mathcal{I}\,]\ .$$

However, due to independence of processing times, we have that $\mathbb{E}\,[P_k \mid \mathcal{I}\,] = \mathbb{E}\,[P_k]$, and therefore

$$\mathbb{E}\,[S_j \mid \mathcal{I}\,] \;\leq\; r'_j \;+ \sum_{k \in H(j) \setminus \{j\}} \mathbb{E}\,[P_k]\ .$$

Consider condition $\mathcal{B}$ and let us denote by $\mathbb{E}\,[x(\mathcal{B})]$ the expected length of the time period until the machine becomes idle for the first time after r'_j. Under the condition $\mathcal{B}$, for any realization p of the processing times, conditioned on $\mathcal{B}$, some job $\ell(p)$ is in process at time r'_j (in fact, $\ell(p)$ might have lower or higher priority than j). Any such job ℓ was available at time $r'_\ell < r'_j$, and by definition of the modified release dates, we therefore know that $\mathbb{E}\,[P_\ell] \leq (1/\alpha) r'_\ell < (1/\alpha) r'_j$ for any such job ℓ. Moreover, letting $t = r'_j - S_\ell$, the expected remaining processing time of such job ℓ, conditioned on the fact that it is indeed in process at time r'_j, is $\mathbb{E}\,[P_\ell - t \mid P_\ell > t\,]$. Due to the assumption

of δ-NBUE processing times, we thus know that

$$\mathbb{E}\left[P_\ell - t \mid P_\ell > t\right] \;\leq\; \delta\,\mathbb{E}\left[P_\ell\right] \;\leq\; (\delta/\alpha)r'_j\,.$$

Therefore, the expected remaining processing time of any job ℓ that might be in process at time r'_j is bounded by $(\delta/\alpha)r'_j$, and thus

$$\mathbb{E}\left[x(\mathcal{B})\right] \;\leq (\delta/\alpha)r'_j\,.$$

Repeating the same argument as above, we can now conclude that

$$\mathbb{E}\left[S_j \mid \mathcal{B}\right] \;\leq\; (1+\delta/\alpha)\,r'_j \;+\; \sum_{k\in H(j)\setminus\{j\}} \mathbb{E}\left[P_k\right]\,. \tag{1}$$

As each of the two conditional expectations $\mathbb{E}\left[S_j \mid \mathcal{I}\right]$ and $\mathbb{E}\left[S_j \mid \mathcal{B}\right]$ is bounded by the right hand side of (1), we obtain that

$$\mathbb{E}\left[S_j\right] \;\leq\; (1+\delta/\alpha)\,r'_j \;+\; \sum_{k\in H(j)\setminus\{j\}} \mathbb{E}\left[P_k\right]\,,$$

and the fact that $\mathbb{E}\left[C_j\right] = \mathbb{E}\left[S_j\right] + \mathbb{E}\left[P_j\right]$ concludes the proof. □

In fact, it is quite straightforward to use Lemma 2 in order to show the following.

Corollary 1. *The α-Shift-*WSEPT *algorithm is a 3-approximation for the single machine problem $1|r_j|\mathbb{E}\left[\sum w_j C_j\right]$, for* NBUE *processing times.*

We just use that $\delta = 1$, and we choose $\alpha = 1$. We skip further details, and note that this matches the best known LP based performance bound derived in [11], which even holds for arbitrary processing time distributions.

The MinIncrease policy for the problem with release dates is now the following. In order to decide on which machine a job should go, we just ignore the release dates, and use the same policy for assigning jobs to machines that we used before in the setting without release dates.

Theorem 3. *Consider the stochastic online scheduling problem on parallel machines with release dates, $\mathrm{P}|r_j|\mathbb{E}\left[\sum w_j\,C_j\right]$. Given that all processing times are δ-NBUE, the modified* MinIncrease *policy is a ρ–approximation, where*

$$\rho = 1 + \max\{1+\delta/\alpha\;,\;\alpha+\delta+(m-1)(\Delta+1)/(2m)\}\,.$$

Here, Δ is such that $\mathrm{Var}[P_j]/\mathbb{E}\left[P_j\right]^2 \leq \Delta$ for all jobs j. In particular, since all processing times are δ-NBUE, we know that $\Delta \leq 2\delta - 1$ in the above performance bound.

Proof. Let i_j be the machine to which job j is assigned. Then, by Lemma 2 we know that

$$\mathbb{E}\left[C_j\right] \leq (1+\frac{\delta}{\alpha})r'_j + \sum_{k\in H(j)\cap M_{i_j}} \mathbb{E}\left[P_k\right]\,, \tag{2}$$

and the expected value of MinIncrease can be bounded by

$$\mathbb{E}\left[Z^{\mathrm{MI}}\right] \le \left(1+\frac{\delta}{\alpha}\right)\sum_j w_j\, r'_j + \sum_j w_j \sum_{k\in H(j)\cap M_{i_j}} \mathbb{E}\left[P_k\right] \tag{3}$$

Using an index rearrangement argument as in the proof of Theorem 2, we can write

$$\sum_j w_j \sum_{k\in H(j)\cap M_{i_j}} \mathbb{E}\left[P_k\right] = \sum_j \left(w_j \sum_{k\in H(j)\cap M_{i_j}(j)} \mathbb{E}\left[P_k\right] + \mathbb{E}\left[P_j\right] \sum_{k\in L(j)\cap M_{i_j}(j)} w_k + w_j \mathbb{E}\left[P_j\right] \right).$$

By definition of MinIncrease, we know that job j is assigned to the machine which minimizes the sums in parenthesis of the right hand side of this equation. Hence, by an averaging argument, we know that

$$\begin{aligned}\sum_j w_j \sum_{k\in H(j)\cap M_{i_j}} \mathbb{E}\left[P_k\right] &\le \sum_j \left(w_j \sum_{k\in H(j),k<j} \frac{\mathbb{E}\left[P_k\right]}{m} + \mathbb{E}\left[P_j\right] \sum_{k\in L(j),k<j} \frac{w_k}{m} + w_j \mathbb{E}\left[P_j\right] \right) \\ &= \sum_j w_j \sum_{k\in H(j)} \frac{\mathbb{E}\left[P_k\right]}{m} \;+\; \frac{m-1}{m} \sum_j w_j \mathbb{E}\left[P_j\right] ,\end{aligned}$$

where the last equality follows from index rearrangement. Plugging this into (2), leads to the following bound on the expected performance of MinIncrease.

$$\mathbb{E}\left[Z^{\mathrm{MI}}\right] \le \left(1+\frac{\delta}{\alpha}\right)\sum_j w_j r'_j + \sum_j w_j \sum_{k\in H(j)} \frac{\mathbb{E}\left[P_k\right]}{m} + \frac{m-1}{m}\sum_j w_j \mathbb{E}\left[P_j\right] .$$

As mentioned before, the relaxed problem without release dates provides a lower bound on the expected optimum with release dates. We therefore can plug into the above inequality the bound of Lemma 1, and obtain

$$\begin{aligned}\mathbb{E}\left[Z^{\mathrm{MI}}\right] \;&\le\; \left(1+\frac{\delta}{\alpha}\right)\sum_j w_j r'_j + \mathbb{E}\left[Z^{\mathrm{OPT}}\right] + \frac{(m-1)(\Delta+1)}{2m}\sum_j w_j \mathbb{E}\left[P_j\right] \\ &=\; \mathbb{E}\left[Z^{\mathrm{OPT}}\right] + \sum_j w_j \left(\left(1+\frac{\delta}{\alpha}\right) r'_j + \frac{(m-1)(\Delta+1)}{2m} \mathbb{E}\left[P_j\right] \right). \end{aligned} \tag{4}$$

By bounding r'_j by $r_j + \alpha\mathbb{E}\left[P_j\right]$, we obtain the following bound on the term in parenthesis of the sum in the right hand side of inequality (4).

$$\begin{aligned}&\left(1+\frac{\delta}{\alpha}\right) r'_j + \frac{(m-1)(\Delta+1)}{2m}\mathbb{E}\left[P_j\right] \\ &\qquad \le \left(1+\frac{\delta}{\alpha}\right) r_j + \left(\alpha+\delta+\frac{(m-1)(\Delta+1)}{2m}\right)\mathbb{E}\left[P_j\right] \\ &\qquad \le \left(r_j + \mathbb{E}\left[P_j\right]\right) \max\left\{ 1+\frac{\delta}{\alpha} \;,\; \alpha+\delta+\frac{(m-1)(\Delta+1)}{2m} \right\} .\end{aligned}$$

The proof is completed by using this inequality in equation (4), and applying the trivial lower bound $\sum_j w_j(r_j + \mathbb{E}[P_j]) \leq \mathbb{E}\left[Z^{\mathrm{OPT}}\right]$ on the expected optimum performance. □

For NBUE processing times, where we can choose $\Delta = \delta = 1$, the approximation ratio is minimal for $\alpha = (\sqrt{5m^2 - 2m + 1} - m + 1)/(2m)$, obtaining a ratio of $2 + (\sqrt{5m^2 - 2m + 1} + m - 1)/(2m)$, which is less than $(5+\sqrt{5})/2 - 1/(2m) \approx 3.62 - 1/(2m)$, improving upon the previously best know approximation ratio of $4 - 1/m$ from [11].

Acknowledgement. The authors would like to thank Rolf H. Möhring for helpful discussions, and Andreas S. Schulz for pointing out a misinterpretation in an earlier version of this paper.

References

1. J. L. Bruno, P. J. Downey, and G. N. Frederickson. Sequencing tasks with exponential service times to minimize the expected flowtime or makespan. *J. ACM*, 28:100–113, 1981.
2. S. Chakrabarti and S. Muthukrishnan. Resource scheduling for parallel database and scientific applications. In *Proc. 8th Ann. ACM Symp. on Parallel Algorithms and Architectures*, pages 329–335, Padua, Italy, 1996.
3. C. Chekuri, R. Johnson, R. Motwani, B. Natarajan, B. Rau, and M. Schlansker. An analysis of profile-driven instruction level parallel scheduling with application to super blocks. In *Proc. 29th IEEE/ACM Int. Symp. on Microarchitecture,* Paris, France, pages 58–69, 1996.
4. M. A. H. Dempster, J. K. Lenstra, and A. H. G. Rinnooy Kan, editors. *Deterministic and Stochastic Scheduling.* D. Reidel Publishing Company, Dordrecht, 1982.
5. W. Eastman, S. Even, and I. Isaacs. Bounds for the optimal scheduling of n jobs on m processors. *Mgmt. Sci.*, 11:268–279, 1964.
6. R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Ann. Discr. Math.*, 5:287–326, 1979.
7. W. J. Hall and J. A. Wellner. Mean residual life. In M. Csörgö, D. A. Dawson, J. N. K. Rao, and A. K. Md. E. Saleh, editors, *Proc. Int. Symp. on Statistics and Related Topics*, pages 169–184, Ottawa, ON, 1981. Amsterdam: North-Holland.
8. E. Koutsoupias and C. H. Papadimitriou. Beyond competitive analysis. *SIAM J. Comp.*, 30:300–317, 2000.
9. N. Megow and A. S. Schulz. On-line scheduling to minimize average completion time revisited. *Oper. Res. Lett.*, 32(5):485–490, 2004.
10. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems I: General strategies. *ZOR - Zeitschrift für Oper. Res.*, 28:193–260, 1984.
11. R. H. Möhring, A. S. Schulz, and M. Uetz. Approximation in stochastic scheduling: the power of LP-based priority policies. *J. ACM*, 46:924–942, 1999.
12. M.H. Rothkopf. Scheduling with random service times. *Mgmt. Sci.*, 12:703–713, 1966.

13. M. Scharbrodt, T. Schickinger, and A. Steger. A new average case analysis for completion time scheduling. In *Proc. 34th Ann. ACM Symp. on the Theory of Computing,* Montréal, QB, pages 170–178, 2002.
14. M. Skutella and M. Uetz. Scheduling precedence-constrained jobs with stochastic processing times on parallel machines. In *Proc. 12th Ann. ACM-SIAM Symp. on Discrete Algorithms*, Washington, DC, pages 589–590, 2001.
15. W. Smith. Various optimizers for single-stage production. *Naval Res. Log.*, 3:59–66, 1956.
16. A. Souza and A. Steger. The expected competitive ratio for weighted completion time scheduling. In *Proc. 21st Symp. on Theoretical Aspects of Computer Science*, Montpellier, France. Lecture Notes in Computer Science 2996, pages 620–631, 2004.
17. M. Uetz. *Algorithms for Deterministic and Stochastic Scheduling.* PhD thesis, Cuvillier Verlag, Göttingen, Germany, 2002.
18. G. Weiss. Approximation results in parallel machines stochastic scheduling. *Ann. Oper. Res.*, 26:195–242, 1990.
19. G. Weiss. Turnpike optimality of Smith's rule in parallel machines stochastic scheduling. *Math. Oper. Res.*, 17:255–270, 1992.
20. Gideon Weiss and Michael Pinedo. Scheduling tasks with exponential service times on non-identical processors to minimize various cost functions. *J. Appl. Prob.*, 17:187–202, 1980.